WHAT IS CLAIMED IS:

1. A processor-implemented method for analyzing operations of an emulated input-output processor (IOP), comprising:

emulating instructions native to a first type of instruction processor on a second-type instruction processor;

executing by an instruction processor emulator an operating system including instructions native to the first type of instruction processor, and including instructions that write input/output (IO) requests to a memory arrangement in response to IO functions invoked by a program;

emulating IOP processing of IO requests with an IOP emulator executable on the second-type processor, including processing IO requests from the memory arrangement and maintaining in the memory arrangement a first set of data structures used in processing the IO requests;

storing state data currently contained in the data structures on a retentive storage device; and

reading state data from the retentive storage responsive to a user input control and displaying the data.

2. The method of claim 1, further comprising:

writing the state data read from retentive storage in a second set of data structures that are of the same type as the first set of data structures

reading selected data from the second set of structures responsive to user-input controls and displaying the selected data.

3. The method of claim 1, wherein the selected data include data that describe each IO request.

4. The method of claim 1, further comprising:

maintaining a cache of data from a storage device by the IOP emulator; and

accumulating statistics describing management of the cache, wherein the selected data include the statistics describing management of the cache.

5. The method of claim 4, wherein the selected data include respective numbers of read requests and write requests.

6. The method of claim 5, wherein the selected data include respective numbers of cache hits and cache misses.

7. The method of claim 6, wherein the selected data include respective numbers of read requests of different sizes and respective numbers of write requests of different sizes.

8. The method of claim 1, wherein the data structures include a first queue of in-process IO requests and a second queue of completed IO requests, and the selected data include data from the first and second queues.

9. The method of claim 1, further comprising:

emulating IOP processing of IO requests with a plurality of IOP emulators executable on the second-type processor, and each IOP emulator maintaining in the memory arrangement respective data structures used in processing IO requests directed to that IOP emulator;

wherein the selected data include data from the data structures associated with an IOP emulator in response to a user-selected IOP emulator.

10. An apparatus for analyzing operations of an emulated input-output processor, comprising:

means for emulating instructions native to a first type of instruction processor on a second-type instruction processor, the instructions including operating system instructions that write input/output (IO) requests to a memory arrangement in response to IO functions invoked by a program;

means for emulating on the second-type processor, IOP processing of IO requests from the memory arrangement and maintaining in the memory arrangement a first set of data structures used in processing the IO requests;

means for storing state data currently contained in the data structures on a retentive storage device; and

means responsive to a user input control for reading state data from the retentive storage and displaying the state data.

11. The apparatus of claim 10, further comprising:

means for maintaining a cache of data from a storage device by the means for emulating IOP processing of IO requests; and

means for accumulating statistics describing management of the cache, wherein the selected data include the statistics describing management of the cache.

12. A processor-implemented method for analyzing operations of an emulated input-output processor, comprising:

emulating instructions native to a first type of instruction processor on a second-type instruction processor;

executing by an instruction processor emulator, an operating system including instructions native to the first type of instruction processor, the operating system including instructions that write input/output (IO) requests to a memory arrangement and read input data from the memory arrangement in response to IO functions invoked by a program;

emulating IOP processing of IO requests with an IOP emulator executable on the second-type processor, including processing IO requests from the memory

arrangement and maintaining in the memory arrangement a first set of data structures used in processing the IO requests;

displaying, via a first tool, selected data from the first set of data structures in response to a user input control;

storing state data currently contained in the data structures on a retentive storage device;

reading state data from the retentive storage responsive to a user input control and writing the state data to a second set of data structures in the memory arrangement, wherein the second set of data structures are identical in structure to the first set of data structures; and

displaying, via the first tool, selected data from the second set of data structures in response to a user input control.

13. The method of claim 12, wherein the selected data from the first and second sets of data structures include data that describe each IO request.

14. The method of claim 12, further comprising:

maintaining a cache of data from a storage device by the IOP emulator; and

accumulating in the first set of data structures statistics that describe management of the cache, wherein the selected data from the first and second sets of data structures include the statistics describing management of the cache.

15. The method of claim 14, wherein the selected data from the first and second sets of data structures include respective numbers of read requests and write requests.

16. The method of claim 15, wherein the selected data include respective numbers of cache hits and cache misses.

17. The method of claim 16, wherein the selected data from the first and second sets of data structures include respective numbers of read requests of different sizes and respective numbers of write requests of different sizes.

18. The method of claim 12, wherein the first and second sets of data structures include a first queue of in-process IO requests and a second queue of completed IO requests, and the selected data from the first and second sets of data structures include data from the first and second queues.

19. The method of claim 12, further comprising:

emulating IOP processing of IO requests with a plurality of IOP emulators executable on the second-type processor, and each IOP emulator maintaining in the memory arrangement respective data structures used in processing IO requests directed to that IOP emulator;

wherein the selected data from the first and second sets of data structures include data from the data structures associated with an IOP emulator in response to a user-selected IOP emulator.